НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

«До захисту допущено»

Завідувач кафедри

_____ Тарасенко В.П.
(підпис) (ініціали, прізвище)

“___” _____2019 р.

Дипломний проект

на здобуття ступеня бакалавра

з напряму підготовки **6.050102 «Комп'ютерна інженерія»**

на тему: «ВЕБ-ОРІЄНТОВАНИЙ ДОДАТОК ДЛЯ МОНІТОРИНГУ ТА АНАЛІЗУ КОМП’ЮТЕРНОЇ МЕРЕЖІ IPV6»

Виконав (-ла): студент (-ка) IV курсу, групи ___КВ-53___
(шифр групи)

_____Сапожніков Богдан Артурович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник _____доц., к.т.н., доц. Щербина О.А._____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з нормоконтролю, доц.каф.СПСКС, к.т.н. Клятченко Я.М.

(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) __ (підпис)

Рецензент професор каф. ОТ, д.т.н., проф. Кулаков Ю.О. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки 6.050102 «Комп’ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Тарасенко В.П.
(підпис) (ініціали, прізвище)
«___» _____2019 р.

ЗАВДАННЯ
на дипломний проект студента
Сапожнікова Богдана Артуровича

1. Тема проекту Веб-орієнтований додаток для моніторингу та аналізу комп’ютерної мережі IPv6.
Керівник проекту: доц., к.т.н., доц. Щербина О.А.,
затверджені наказом по університету від «22» травня 2019 р. №1330-С

2. Термін подання студентом проекту “___” _____ 2019 р.

3. Вихідні дані до проекту : див. технічне завдання.

4. Зміст пояснювальної записки: аналіз існуючих рішень, обґрунтування теми, обґрунтування вибору методів реалізації, розробка компонентів системи, аналіз роботи програмного продукту

5. Перелік графічного матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо) : «Система моніторингу мережі. Схема структурна.», «Основний алгоритм роботи.», «Основний алгоритм обробки запитів користувача.», «Алгоритм роботи мережного модуля.»

6. Консультанти розділів проекту*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Клятченко Я.М., доц. каф. СПіСКС, к.т.н.		

7. Дата видачі завдання: “___” _____ 2019 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1.	Вивчення літератури за тематикою проекту	21.02.2019	
2.	Розробка та узгодження технічного завдання	01.03.2019	
3.	Аналіз існуючих рішень	10.03.2019	
4.	Створення структури Web-додатку	15.03.2019	
5.	Вибір середовища розробки	20.03.2019	
6.	Розробка веб-серверу	25.03.2019	
7.	Розробка додатку для роботи з мережею	11.04.2019	
8.	Розробка додатку для аналізу даних	25.04.2019	
9.	Відлагодження програмного продукту	05.05.2019	
10.	Підготовка пояснювальної записки	10.05.2019	
11.	Оформлення матеріалів проекту	12.05.2019	
12.	Попередній показ дипломного проекту на кафедрі	28.05.2019	

Студент _____ Сапожніков Б.А.
(підпис)

Керівник проекту _____ Щербина О.А.
(підпис)

* Консультантом не може бути зазначено керівника дипломного проекту.

АНОТАЦІЯ

Кваліфікаційна робота включає пояснювальну записку (___ с., ___ рис., ___ табл., ___ додатки).

В бакалаврському проекті розроблено Web-орієнтований додаток для моніторингу та аналізу комп'ютерної мережі IPv6, що дозволяє на практиці засвоїти знання з курсу «Комп'ютерні мережі».

Розроблена система призначена для реалізації функцій моніторингу та контролю якості з'єднання у IP мережі з використанням протоколу шостої версії.

Даний програмний комплекс орієнтований на підтримку лабораторного курсу «Комп'ютерні мережі». До нього входять наступні елементи:

- веб-сервер, що приймає вхідні дані та передає їх на модулі обробки, а також зберігає та відображає результати роботи;
- програмні модулі для роботи з мережею та аналізу даних;
- простий веб-інтерфейс, що забезпечує можливість використовувати веб-додаток звичайному користувачу.

Розроблена програма дозволяє на практиці ознайомитись з роботою IP-протоколів зокрема протоколу шостої версії та дослідити параметри з'єднання в мережі.

Ключові слова: протокол IP, IPv6, аналіз даних, веб-додаток, мережа IPv6, параметри з'єднання.

ABSTRACT

The qualifying work includes an explanatory note (___ p., ___pics, ___ tables, ___ annexes).

The bachelor's project has developed a Web-based application for monitoring and analysis of the IPv6 computer network, which allows to practice knowledge of the course "Computer Networks".

The developed system is intended for realization of functions of monitoring and control of connection quality in IP networks using the protocol of the sixth version.

This software package is aimed at supporting the laboratory course "Computer Networks". It includes the following elements:

- a web server that accepts incoming data and transmits it to the processing unit, as well as stores and displays the results of the work;

- software modules for network operation and data analysis;

- a simple web-based interface that provides the ability to use the web application to the regular user.

The developed program allows you to familiarize yourself with the work of IP-protocols, in particular the protocol of the sixth version, and to investigate network connection parameters.

Keywords: IP protocol, IPv6, data analysis, web application, IPv6 network, connection settings.

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
			Документація загальна			
			Новорозроблена			
	А4	ІАЛЦ.467100.002 ТЗ	Веб-орієнтований	4		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPV6.			
			Технічне завдання			
	А4	ІАЛЦ.467100.003 ТП	Веб-орієнтований	3		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Відомість технічного			
			проекту			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467100.001 ОА		
Розробив		Сапожніков Б.А			Веб-орієнтований додаток для моніторингу та аналізу комп’ютерної мережі IPv6 Опис альбому	Літ.	Аркуш 1 / Аркушів 4
Перевірив		Щербина О.А.					
Н. контроль		Клятченко Я.М.				КПІ ім. Ігоря Сікорського ФПМ КВ-53	
Затвердив		Тарасенко В.П.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467100.004 ПЗ	Веб-орієнтований	50		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Пояснювальна записка			
	А4	ІАЛЦ.467100.005 Е1	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Схема структурна.			
	А4	ІАЛЦ.467100.006 Д1	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Основний алгоритм			
			роботи.			

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467100.007 Д2	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Основний алгоритм			
			Обробки запитів			
			користувача.			
	А4	ІАЛЦ.467100.008 Д3	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Алгоритм роботи			
			Мережного модуля.			

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
		Диск CD-ROM	Текст ПЗ. Тексти програм.	1		
			Графічний матеріал			

ЗМІСТ

стор.

					ІАЛЦ.467100.002 ТЗ		
Зм.	Арк.	№ докум.	Підп.	Дата			
Розроб.		Сапожніков Б.А.			Веб-орієнтований додаток для моніторингу та аналізу комп'ютерної мережі IPv6 Технічне завдання	Літ.	Аркуш 1 / Аркушів 4
Перевір.		Щербина О.А.					
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського ФПМ КВ-53	
Затв.		Тарасенко В.П.					

1. НАЙМЕНУВАННЯ І ОБЛАСТЬ ЗАСТОСУВАННЯ

Найменування роботи – «Web-орієнтований додаток для моніторингу та аналізу сегменту комп’ютерної мережі IPv6».

Область застосування: у рамках тестування якості обслуговування сегменту мережі IPv6.

2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на дипломне проектування першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп’ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

3. ЦІЛЬ І ПРИЗНАЧЕННЯ РОБОТИ

Метою даного проекту є створення програми для моніторингу та аналізу інформації про стан комп’ютерної мережі IPv6.

4. ДЖЕРЕЛА РОБОТИ

Джерелами роботи є конспект лекцій з курсу «Комп’ютерні мережі», науково-технічна література з теорії комп’ютерних мереж, статті у мережі Інтернет.

5. ТЕХНІЧНІ ВИМОГИ

5.1 Вимоги до Web-орієнтованого додатку.

- На вхід приймає IPv6 адресу або доменне ім’я для тестування.
- На основі аналізу вхідних даних формуються тестові пакети.
- В разі виникнення помилок при вводі даних або доступі до віддаленого серверу, має бути надане відповідне повідомлення.
- Виходом програми є дані моніторингу та аналізу стану сегменту мережі IPv6 в якому знаходиться дана адреса.

5.2 Вимоги до програмного забезпечення.

- Операційна система Windows, Linux.
- PyCharm 2019.1.
- Браузер.

5.3 Вимоги до апаратного забезпечення.

- Монітор.
- Комп’ютер.
- Підключення до мережі Інтернет.

6. ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів роботи та питань, які мають бути розроблені відповідно до завдання	Термін виконання
1.	Видача завдання на дипломне проектування	19.11.2018
2.	Вивчення літератури за тематикою проекту	10.01.2019
3.	Розробка та узгодження технічного завдання	10.02.2019
4.	Аналіз існуючих рішень	25.02.2019
5.	Створення структури Web-додатку	20.03.2019
6.	Вибір середовища розробки	25.03.2019
7.	Розробка веб-серверу	11.04.2019
8.	Розробка модуля для роботи з мережею	25.04.2019
9.	Розробка модуля для аналізу даних	05.05.2019
10.	Відлагодження програмного продукту	10.05.2019
11.	Підготовка пояснювальної записки	12.05.2019
12.	Оформлення матеріалів проекту	26.05.2019
13.	Попередній показ дипломного проекту на кафедрі	28.05.2019

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
			Документація загальна			
			Новорозроблена			
	А4	ІАЛЦ.467100.004 ПЗ	Веб-орієнтований	50		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Пояснювальна записка			
	А4	ІАЛЦ.467100.005 Е1	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Схема структурна.			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467100.003 ВП		
Розробив		Сапожніков Б.А.			Веб-орієнтований додаток для моніторингу та аналізу комп’ютерної мережі IPv6 Відомість технічного проекту	Літ.	Аркуш 1 / Аркушів 3
Перевірив		Щербина О.А.					
Н. контроль		Клятченко Я.М.				КПІ ім. Ігоря Сікорського ФПМ КВ-53	
Затвердив		Тарасенко В.П.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467100.006 Д1	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Основний алгоритм			
			роботи.			
	А4	ІАЛЦ.467100.007 Д2	Веб-орієнтований			
			додаток для моніторингу			
			та аналізу комп’ютерної	1		
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Основний алгоритм			
			обробки запитів			
			користувача.			

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467100.008 ДЗ	Веб-орієнтований	1		
			додаток для моніторингу			
			та аналізу комп’ютерної			
			мережі IPv6.			
			Система моніторингу			
			мережі.			
			Алгоритм роботи			
			мережного модуля.			
		Диск CD-ROM	Текст ПЗ. Тексти програм.	1		
			Графічний матеріал			

ЗМІСТ

стор.

					ІАЛЦ.467100.004 ПЗ		
Зм.	Лист	№ докум.	Підп.	Дата			
Розробив		Сапожніков Б.А.			Веб-орієнтований додаток для моніторингу та аналізу комп’ютерної мережі IPv6	Літ.	Аркуш 1 / Аркушів 50
Перев.		Щербина О.А.					
Н. контр.		Клятченко Я.М.			Пояснювальна записка	КПІ ім. Ігоря Сікорського ФПМ КВ-53	
Затвер.		Тарасенко В.П.					

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ARP	Address Resolution Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6
EUI	Extended Unique Identifier
FTP	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICMPv6	Internet Control Message Protocol. Internet Protocol Version 6
IMAP	Internet Message Access Protocol
IPv4	Internet protocol version 4
IPv6	Internet protocol version 6
IRDP	Internet Router Discovery Protocol
ISP	Internet Service Provider
LAN	Local Area Network
NAT	Network Address Translation
ND	Neighbor Discovery
NMS	Network Management Station
OSI	Модель взаємоз’єднання відкритих систем
QoS	Quality of service
RA	Router advertisement
RDP	Remote Desktop Protocol
SMTP	Simple Mail Transfer Protocol
TTL	Time To Live
VPN	Virtual Private Network
КМ	Комп’ютерна мережа
Хост	Кінцевий користувач у мережі
Шлюз	Роутер, що з’єднує декілька мереж

ВСТУП

Комп'ютерні мережі - це технологія, яка створює зв'язок між комп'ютерами та іншими пристроями. Також їх можна визначити як обмін мережевими пакетами між обчислювальними машинами\пристроями по всьому світу за допомогою ліній передачі даних, таких як дротяні кабелі, оптичні волокна тощо.

Мережі є дуже важливим фактором для сьогоднішньої глобалізації. Одним з ключових факторів розвитку інформаційних технологій у світі є мережа, як спосіб передачі даних між пристроями, оскільки передача інформації - це не тільки спосіб зв’язку, а й утворення цілісної системи, що дозволяє людям почувати себе комфортно. Телебачення, мобільні телефони, телефони стаціонарні, що в найближче десятиліття обіцяють стати історією, наші комп’ютери та планшети, розумні годинники та десятки інших звичних речей – це все частини однієї системи, створеної для задоволення людських потреб. А будь-яка система потребує комунікації між її елементами, саме цей зв’язок і прийнято називати мережею.

Перші комп’ютерні мережі створювалися, як спосіб розподілити реалізацію складних обчислень на декілька комп’ютерів для економії часу. Згодом їх почали використовувати для швидкого обміну інформацією на великих відстанях. Звичайно ж у 60-х роках минулого століття воно мало нагадувало сучасну мережу Інтернет, що зараз дозволе нам спілкуватися один з один, не зважаючи на відстань, знаходити та використовувати знання та інформацію зі всього світу. Тоді сьогоднішні реалії виглядали фантастикою. Швидкість та якість передачі інформації сильно обмежувалися продуктивністю обладнання, ненадійністю і недосконалістю фізичних каналів зв’язку. Проте це все одно було проривом, що в майбутньому дозволив створити світ таким, яким ми бачимо його зараз.

Мережа Інтернет – це надпотужний засіб комунікації та обміну знаннями й технологіями. Саме вона дозволила суспільству розвиватися

такими темпами. Для прикладу, мережі розподілених обчислень допомагали розраховувати параметри перших космічних шаттлів і супутників. Зараз Інтернет дозволяє людям по всій планеті навчатися майже будь чому. Навіть не просто навчитися, а й отримати повноцінну професію і працювати по ній за допомогою того ж інтернету.

Саме тому дуже важливо бути впевненим, що необхідні людям сегменти мережі працюють без збоїв і будь-хто в будь який момент можемо отримати доступ до інформації на віддаленому сервері.

Моніторинг, як комп'ютерних мереж загалом, так і зокрема мереж, що працюють на основі протоколу IPv6, є важливим інструментом у руках інтернет-провайдерів для виявлення та усунення несправностей у їх мережах. Завдяки чому більша частина людства має безперебійний доступу до мережі Інтернет.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ, ОБҐРУНТУВАННЯ ТЕМИ ДИПЛОМНОГО ПРОЕКТУ

1.1 Загальні відомості та можливості мереж

Основна мета комп'ютерних мереж - полегшити комунікацію. Мережа дозволяє користувачеві миттєво з'єднуватися з іншим користувачем або мережею, а також надсилати та отримувати дані. Це дозволяє віддаленим користувачам з'єднуватися один з одним за допомогою відеоконференцій, віртуальних зустрічей та цифрових повідомлень електронної пошти.

Комп'ютерні мережі надають доступ до онлайн-бібліотек, журналів, електронних газет, чатів, веб-сайтів соціальних мереж, поштових клієнтів і всесвітньої мережі. Користувачі можуть скористатися онлайн-бронювання для театрів, ресторанів, готелів, поїздів і літаків. Вони можуть робити покупки і здійснювати банківські операції не виходячи зі своїх будинків.

Спільне використання ресурсів.

Комп'ютерні мережі дозволяють користувачам обмінюватися файлами. Вони широко використовуються в організаціях для зменшення витрат і спрощення спільного використання ресурсів. Один принтер, приєднаний до невеликої локальної мережі (LAN), може ефективно обслуговувати запити друку всіх користувачів ПК в цій мережі. Користувачі можуть подібно використовувати інші мережні пристрої та обладнання, такі як модеми, факси, жорсткі диски і тд.

Мережі дозволяють користувачам спільно використовувати програми та файли. Програмне забезпечення для обробки текстів, відео, фотографії, аудіофайли, програмне забезпечення для відстежування стану проекту та інші подібні програми можуть використовуватися онлайн. Користувачі можуть також отримувати доступ для зберігання та використання даних на жорсткому диску мережевого сервера.

Комп'ютерні мережі централізують підтримку, адміністрування та завдання підтримки мережі. Технічний персонал керує всіма вузлами мережі, надає допомогу та усуває помилки мережевого обладнання та програмного забезпечення. Мережеві адміністратори забезпечують цілісність даних і розробляють системи для підтримки надійності передачі інформації через мережу. Вони відповідають за забезпечення високого рівня антивірусного, антишпигунського та брандмауерного програмного забезпечення для користувачів у мережі. На відміну від автономної системи, комп'ютер, підключений до мережі, може повністю управлятися і адмініструватись централізованим сервером, який приймає всі запити від користувачів і обслуговує їх потреби.

1.2 Стандартні терміни в комп'ютерних мережах.

1) Вузли в комп'ютерних мережах означають будь-які обчислювальні пристрої, такі як комп'ютери, мобільні телефони, планшети тощо, які намагаються передавати та отримувати мережеві пакети через мережу до іншого подібного пристрою.

2) Мережеві пакети є нічим іншим, як інформацією або одиницями даних, які вихідний вузол хоче відправити/отримати в/з вузла призначення.

3) IPv4 адреси складаються з 32 біт (чотири байти). Прикладом адреси IPv4 буде 91.198.174.192. Зараз IP адреси четвертої версії – найпоширеніший спосіб для ідентифікації пристроїв, під’єднаних до мереж, у світі.

4) IPv6 адреси досить нові для світу і складаються з восьми шістнадцяткових номерів (128 біт), розділених «:». Прикладом адреси IPv6 буде «2001: 0cb8: 85a3: 0000: 0000: 8a2e: 0370: 7334». На даний момент їх використання обмежено недосконалістю більшості мереж, які створювалися для роботи виключно з четвертою версію протоколу IP. Тобто передача

пакетів шостої версії протоколу через застарілі мережі потребує спеціальних перетворень пакетів.

5) Маршрутизатори - це апаратний компонент, який виконує маршрутизацію пакетів. Маршрутизація – це процес визначення маршруту передачі пакета по мережі. Жоден комп'ютер не «знає», де розташовані інші комп'ютери. Маршрутизатор ідентифікує адресу вузла призначення, до якого повинен бути надісланий мережевий пакет, і пересилає його на потрібну адресу. Маршрутизатори мають специфічний «протокол маршрутизації», який визначає формат, в якому вони обмінюються даними з іншим маршрутизатором або мережевими вузлами. Іншими словами, протокол маршрутизації визначає, як маршрутизатори спілкуються один з одним. Маршрутизатори створюють «таблицю маршрутизації», яка визначає оптимальні шляхи, які необхідно прийняти в мережі при передачі пакетів[1].

1.3 Модель взаємоз’єднання відкритих систем (OSI)

Вона визначає мережевий фреймворк для реалізації протоколів у рівнях, при цьому управління передається від одного рівня до іншого. Це концептуально поділяє архітектуру комп'ютерної мережі на 7 рівнів в логічній прогресії. Нижні рівні мають справу з електричними сигналами, блоками двійкових даних і маршрутизацією цих даних по мережах. Більш високі рівні охоплюють мережеві запити та відповіді, подання даних і мережеві протоколи, що видно з точки зору користувача. Модель OSI спочатку була задумана як стандартна архітектура для побудови мережевих систем, і багато сучасних мережевих технологій сьогодні відображають розбиту на рівні структуру OSI (рис 1.1) [2].

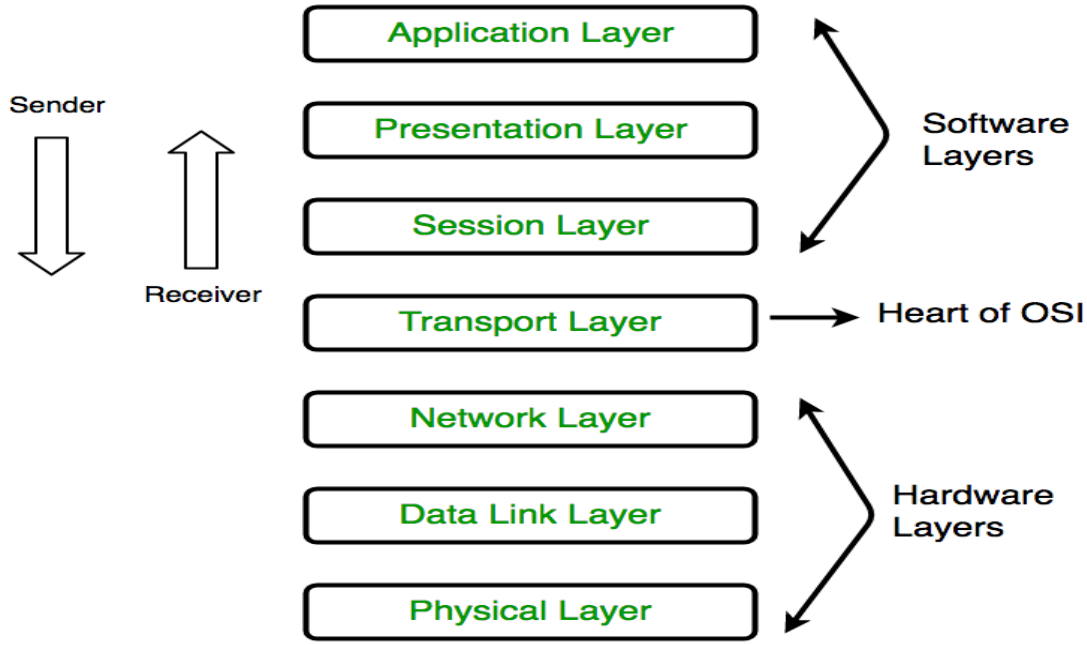

Рисунок 1.1 – Структурна модель OSI

Фізичний рівень

Перший - фізичний рівень моделі OSI відповідає за остаточну передачу бітів цифрових даних відправляючого (вихідного) пристрою через мережеві комунікаційні носії до фізичного рівня приймаючого (кінцевого) пристрою. Він відповідає за фактичне фізичне з'єднання між пристроями.

Канальний рівень (Data Link Layer)

Він відповідає за доставку даних з одного вузла мережі в інший. При отриманні даних з фізичного рівня, рівень передачі даних перевіряє на фізичні помилки. Рівень Data Link також управляє схемами фізичної адресації, такими як MAC-адреси для мереж Ethernet, контролюючи доступ будь-яких різних мережевих пристроїв до фізичного середовища [2].

Мережевий рівень

Мережевий рівень працює для передачі даних від одного хоста до іншого, розташованого в різних мережах. Він також відповідає за маршрутизацію пакетів, тобто вибирає найкоротший шлях для передачі пакета з числа доступних маршрутів. IP-адреса відправника та одержувача

розміщується в заголовку мережевим рівнем. Мережний рівень також керує відображенням між логічними адресами та фізичними адресами. У мережах IP це відображення здійснюється за допомогою протоколу ARP.

Транспортний рівень

Транспортний рівень надає дані через мережні підключення. TCP - найпоширеніший приклад мережевого протоколу транспортного четвертого рівня. Дані в транспортному шарі називаються сегментами. Транспортний рівень також забезпечує підтвердження успішної передачі даних і повторно передає дані, якщо знайдена помилка. Різні транспортні протоколи можуть підтримувати ряд додаткових можливостей [2].

Сеансовий рівень

Цей рівень відповідає за встановлення з'єднання, підтримку сеансів, аутентифікацію і також забезпечує безпеку.

Сеансовий рівень дозволяє встановлювати, використовувати і припиняти з'єднання двох процесів. Цей рівень дозволяє процесу додавати контрольні точки, які вважаються точками синхронізації даних. Ці точки синхронізації допомагають ідентифікувати помилку, щоб дані були повторно синхронізовані належним чином. Контролер діалогу: сеансовий рівень дозволяє запустити зв'язок між двома системами в напівдуплексному або повнодуплексному режимі [2].

Рівень презентації

На шостому рівні виконується синтаксична обробка даних повідомлень, таких як перетворення формату і шифрування/дешифрування, необхідні для підтримки рівня програми над ним. Цей рівень також називається рівнем представлення даних. Дані з прикладного рівня перетворюються відповідно до необхідного формату для передачі по мережі [2].

Прикладний (програмний) рівень

Цей рівень постачає мережеві послуги для кінцевих користувачів. Мережеві послуги - це, як правило, протоколи, які працюють з даними

користувача. Наприклад, у веб-браузері HTTP-протоколу пакує дані, необхідні для надсилання та отримання вмісту веб-сторінки [2].

1.4 IP Мережі(Internet Protocol Network)

Комп’ютерні мережі, що працюють на основі технології IP забезпечують зв'язок між обчислювальними пристроями. Щоб правильно взаємодіяти, всім комп'ютерам (хостам) у мережі потрібно використовувати однакові комунікаційні протоколи. IP Мережа - це мережа комп'ютерів, що використовує Internet Protocol як протокол зв'язку. Всі комп'ютери в такій мережі повинні мати IP-адресу, яка унікально ідентифікує кожен окремий хост. Мережа Інтернет-протоколу (IP-мережа) - це група хостів, які мають спільне фізичне з'єднання і використовують Інтернет-протокол для комунікації на рівні мережі.

1.5 Структура пакетів IP

У пакетах TCP/IPv4 є заголовок пакета TCP (або UDP), потім заголовок пакету IPv4, потім пакетні дані. Кожен заголовок (рис. 1.2) є структурованим набором даних, включаючи такі речі, як адреса джерела і адреса призначення. Найбільші зміни від IPv4 до IPv6 пов'язані з новою та вдосконаленою архітектурою заголовка IP-пакетів у IPv6(рис. 1.3).

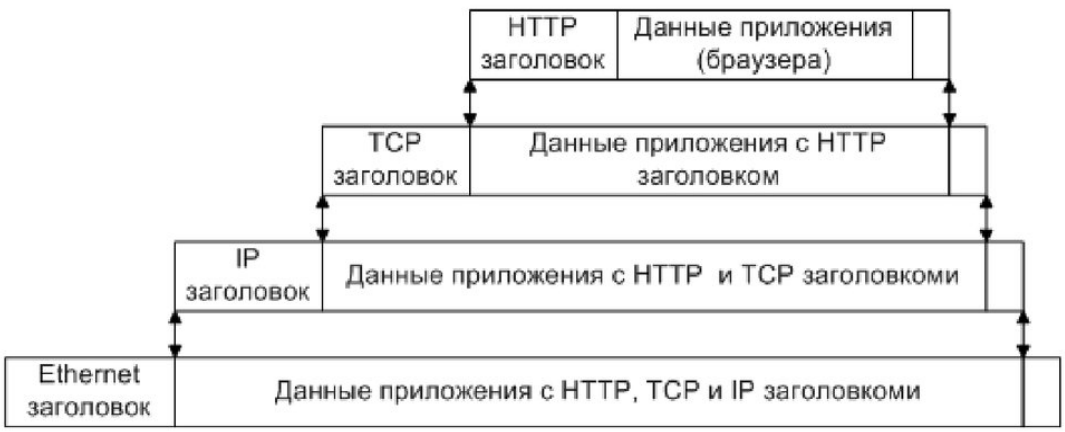

Рисунок 1.2 – Стек заголовків в IP мережах

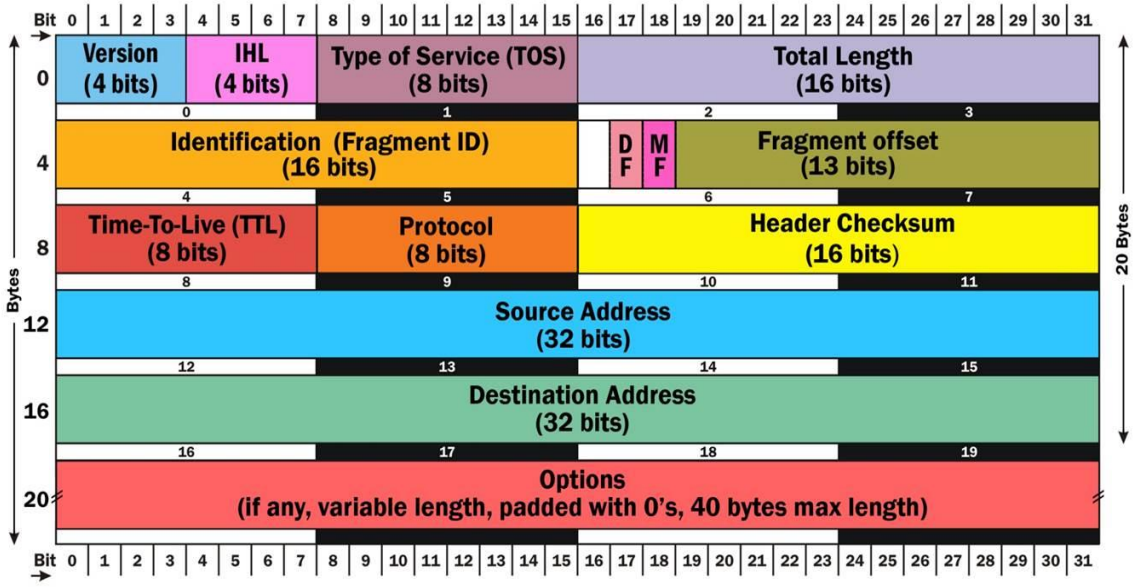

Рисунок 1.3 – Структура заголовку IPv4

Поле версії IP (4 біти) містить значення 4, яке в двійковому форматі має значення «0100».

Поле «Довжина заголовка» (4 біти) вказує на довжину заголовка у 32-бітових «словах». Мінімальне значення - "5", яке має становити 160 біт або 20 байтів. Максимальна довжина становить 15, що становить 480 біт або 60 байт. Якщо пропустити таку кількість слів від початку пакета, то там починаються дані (це називається "зміщенням" даних).

Поле Type of Service (8 біт) визначено в RFC 2474, "Визначення поля диференційованих послуг (поле DS) в заголовках IPv4 і IPv6", грудень 1998 року. QoS [6], передбачає управління пропускною спроможністю, наприклад, можна додати VoIP-з'єднанням більш високий пріоритет, ніж завантаженню відео.

Поле Total Length (16 біт) містить загальну довжину пакета, включаючи заголовок пакета, в байтах. Мінімальна довжина становить 20 (20 байт заголовка плюс 0 байт даних), а максимум - 65535 байтів (оскільки доступно лише 16 біт). Всі мережні системи повинні обробляти пакети щонайменше 576

байтів, але більш типовий розмір пакета становить 1508 байт. За допомогою IPv4 деякі пристрої (наприклад, маршрутизатори) можуть фрагментувати пакети (розбивати їх на декілька менших пакетів), якщо потрібно, щоб вони пройшли через частину мережі, яка не може обробляти великі пакети. Пакети, які є фрагментованими, повинні бути зібрані на іншому кінці. Фрагментація та повторна збірка є однією з найгірших властивостей IPv4, вона була значно покращена в IPv6.

Ідентифікаційне поле (ідентифікатор фрагмента) (16 біт) містить у собі унікальний ідентифікатор пакету, що допомагає при відновленні цілісного повідомлення після фрагментації. В пакеті IPv6 фрагментація не здійснюється проміжними вузлами, тому всі поля заголовка, пов'язані з фрагментацією, більше не потрібні.

Наступні три біти - це прапори, пов'язані з фрагментацією. Перший біт зарезервований і повинен бути нульовим. Наступний біт - прапор DF. Якщо встановлено DF, пакет не може бути фрагментованим (тому, якщо такий пакет досягає частини мережі, яка не може обробляти таку велику, що пакет скидається). Третій біт - прапор MF (More Fragments). Якщо встановлено MF, то будуть ще фрагменти, а цей фрагмент не останній. Нефрагментовані пакети, звичайно, мають прапор MF, встановлений в нуль.

Поле зміщення фрагмента (13 біт) використовується для повторної збірки фрагментованих пакетів. Він вимірюється в 8-байтових блоках. Перший фрагмент набору має зміщення 0.

Поле Time To Live (TTL) (8 біт) запобігає безперервному переміщенню пакетів у мережі. Спочатку він був розрахований на тривалість життя в секундах, але він почав реалізовуватись як "hop by hop". Це означає, що кожен раз, коли пакет передається через комутатор або маршрутизатор, TTL зменшується на одиницю. Якщо він досягає нуля, пакет знищується. Як правило, якщо це відбувається, повідомлення ICMPv4 ("час перевищено") повертається відправнику пакетів.

Поле Протокол (8 біт) визначає тип даних, що знаходяться на початку поля. Номери протоколів не є номерами портів.

Поле контрольної суми заголовка (16 біт). 16-бітне доповнення додаткової суми всіх 16-бітних слів у заголовку. При обчисленні поле контрольної суми береться як нуль. Щоб перевірити контрольну суму, необхідно додати всі 16 бітові слова в заголовку разом, включаючи передану контрольну суму. Результат повинен бути 0. Якщо буде отримано будь-яке інше значення, то принаймні один біт у пакеті пошкоджений. Оскільки TTL зменшується на одиницю на кожному стрибку, контрольна сума заголовка IP повинна бути перерахована при кожному переході. Контрольна сума заголовка IP була вилучена в IPv6.

Поле джерела адреси (32 біта) містить адресу IPv4 відправника (може бути змінено NAT).

Поле Адреса призначення (32 біта) містить адресу IPv4 одержувача (може бути змінений NAT в пакеті відповідей).

1.6 Internet Protocol version 6 (IPv6)

IPv6 вводиться і реалізується для подолання очікуваного дефіциту IP-адресації у IPv4. Він призначений для підтримки постійно зростаючої кількості користувачів і функціональності в Інтернеті. В даний час більшість мереж перебуває в перехідному періоді, використовуючи одночасно IPv4 і IPv6. Зі створенням нових пристроїв, які можуть бути приєднані до Інтернету, неминуче зросте попит на нові адреси Інтернет-протоколів.

Крім IP-адрес, до яких можна звертатися, є багато нових можливостей і вдосконалень, які були включені в IPv6. Це новий протокол третього рівня OSI, а отже, й виникає необхідність розробки нового механізму співіснування з поточним стеком TCP/IP. Нові версії відповідних протоколів у стеку TCP/IP, таких як ICMP6, вже розроблені.

IPv4 був розроблений вже давно, приблизно в січні 1980 року, і з моменту його впровадження з'явилося багато запитів на додаткові адреси та розширення можливостей. Основною зміною в IPv6 стала зміна формату заголовка, включаючи збільшення розміру адреси з 32 біт до 128 біт. Оскільки третій рівень відповідає за комплексну пакетну передачу, використовуючи маршрутизацію пакетів на основі адрес, він повинен включати нові адреси IPv6 відправника і отримувача, подібні до IPv4 [7].

Нові масові ринки вимагають нового набору вимог, які не були очевидні на ранніх стадіях розгортання IPv4. Пристрої, що знаходять на кінцевих вузлах Інтернет комунікації, варіюються від суперкомп'ютерів до будь-яких побутових приладів, підключених до мережі задля спрощеного управління. Більшість з них приєднані до локальних мереж.

Сучасні умови споживчого ринку вимагають рішень, які є надійними, простими у використанні та з низькою вартістю. Використання смарт-карти як ідентифікації людини з персональним набором служб є іншим потенційним кандидатом для призначення IP-адрес. У цьому випадку кожна людина в світі матиме в майбутньому персональну IP-адресу, вбудовану в смарт-карту або подібний пристрій. Глобальна інтернет-маршрутизація на основі 32-бітних адрес IPv4 стає все більш напруженою. Адреси IPv4 не забезпечують достатньої гнучкості для побудови ефективних ієрархій, які можна об'єднати. Розгортання міждоменної маршрутизації (CIDR) збільшує термін служби маршрутизації IPv4 на декілька років, однак надовго цього не вистачить.

1.7 Розширені можливості маршрутизації та адресації у IPv6

IPv6 збільшує розмір IP-адреси з 32 біт до 128 біт, підтримуючи набагато більшу кількість адресних вузлів і більше рівнів ієрархії адресації, а також більш просту автоматичну конфігурацію адрес. Масштабованість багатоадресної маршрутизації покращується шляхом додавання поля сфери до адрес для багатоадресної передачі. Не маючи прихованих мереж і хостів, всі

хости можуть бути доступними і бути серверами, що дозволяють досягти глобальної досяжності. Використання 64 бітів для адресного поля є гарантією унікальності.

Визначається новий тип адреси, що називається адресою anycast, для ідентифікації наборів вузлів, в яких пакет, надісланий до адреси anycast, доставляється до одного з вузлів групи. Використання адрес anycast в полі джерела IPv6 дозволяє вузлам керувати транспортним потоком. Обов'язкові функції включають безпеку, таку як IP Security (IPSec).

Немає броадкаст передач, що забезпечує ефективне використання мережі та меншу кількість переривань у мережевих адаптерах. IPv6 надає великий вибір адрес багатоадресної передачі з кількома варіантами визначення масштабу.

Спрощення формату заголовка

Деякі поля заголовка IPv4 були видалені або зроблені необов'язковими, щоб зменшити витрати на обробку пакетів і зберегти максимальну швидкість смуги пропускання IPv6, незважаючи на збільшення розміру адрес. Незважаючи на те, що адреси IPv6 є в чотири рази більшими, ніж адреси IPv4, заголовок IPv6 лише в два рази перевищує розмір заголовка IPv4.

Удосконалення опцій

Параметри IPv6 розміщуються в окремих заголовках, розташованих між заголовком IPv6 і заголовком транспортного рівня. Зміни в кодуванні параметрів заголовка IP дозволяють більш ефективно пересилати дані. Реалізують менш жорсткі обмеження на довжину опцій і більшу гнучкість для введення нових опцій у майбутньому. Зменшення кількості полів, наприклад, видалення контрольної суми, забезпечує високу ефективність маршрутизації, продуктивність, швидкість пересилання і розширюваність заголовка.

Якість обслуговування (QoS)

Додано новий потенціал потокової мітки, для визначення пакетів, що належать до певних потоків трафіку, для яких відправник вимагає спеціальної

обробки, наприклад, якість обслуговування або передача потокового трафіку в реальному часі.

Аутентифікація та конфіденційність

IPv6 включає визначення розширень, які забезпечують підтримку аутентифікації, цілісності даних та конфіденційності. Це включено як базовий елемент.

1.8 Заголовок пакетів IPv6

Без додаткових заголовків його розмір становить 40 байт. Він, хоч і займає значно більше місця, ніж заголовок четвертої версії, але за рахунок меншої загальної кількості полів та меншого числа полів (рис. 1.4), що потребують обробки на кожному з вузлів, по швидкості обробки він значно перевищує заголовок четвертої версії.

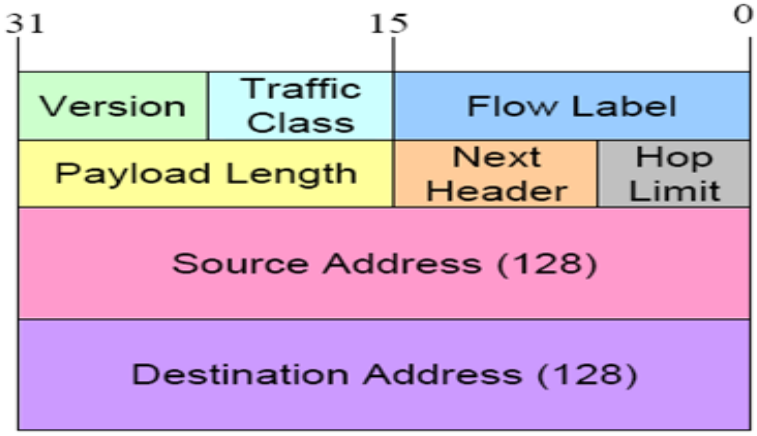

Рисунок 1.4 – Структура заголовку IPv6

Розглянемо призначення кожного з полів заголовку шостої версії протоколу.

Версія -4-розрядний номер версії Інтернет-протоколу = 6.

Клас трафіку - 8-бітове поле класу трафіку. Визначає різні класи або пріоритети (diffserv).

Мітка потоку - 20-бітова мітка потоку. Використовується вихідним вузлом для маркування послідовностей пакетів

Довжина корисного навантаження -16-бітове ціле число без знака, тобто залишок пакета, що слідує за цим заголовком IPv6, в октетах. Будь-які заголовки розширень, вважаються частиною корисного навантаження, тобто включені у довжину.

Наступний заголовок - 8-бітний блок. Ідентифікує тип заголовка, який розташований одразу за заголовком IPv6. Використовує ті ж значення, що і поле протоколу IPv4. Використовується для ідентифікації інкапсульованого протоколу - TCP, UDP, ESP, AH (конфіденційність і аутентифікація в IPsec), ICMPv6 та інші розширення.

Hop Limit - 8-бітове ціле число без знака. Кожен вузол, який пересилає пакет, зменшується на 1. Пакет знищується, якщо Hop Limit зменшується до нуля. Аналог TTL в IPv4.

Адреса джерела - 128-бітна адреса пристрою ініціатора пакету.

Адреса призначення - 128-бітна адреса одержувача пакету (можливо, не кінцевий одержувач, якщо присутній заголовок маршрутизації)

Заголовки розширень - У IPv6, необов'язкова інформація кодується в окремих заголовках, які можуть бути розміщені між заголовком IPv6 і заголовком верхнього рівня в пакеті. Існує невелика кількість таких заголовків-розширень, кожна з яких ідентифікована окремим значенням наступного заголовка. Пакет IPv6 може містити нуль, один або більше заголовків розширень, кожен з яких ідентифікований полем Наступний заголовок полем попереднього заголовка (рис. 1.5).

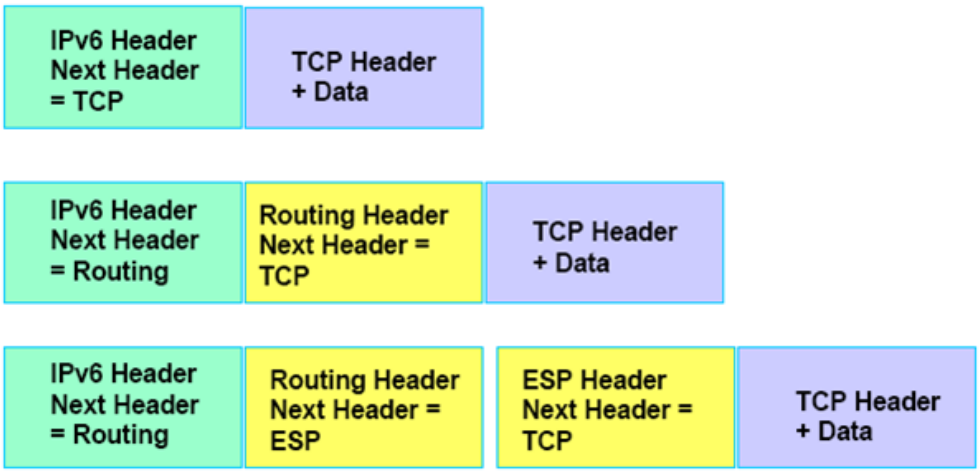

Рисунок 1.5 – Структурна схема розширених заголовків

Кожен додатковий заголовок має довжину 8 октетів (1 октет = 8 біт), для того, щоб зберегти 8-октетне вирівнювання для наступних заголовків.

Повна реалізація IPv6 включає реалізацію наступних заголовків розширень.

- Заголовок IPv6.
- Заголовок Hop-by-Hop Options.
- Заголовок Параметри призначення (коли використовується заголовок маршрутизації).
- Заголовок маршрутизації.
- Заголовок фрагмента.
- Заголовок автентифікації.
- Інкапсулювання заголовка корисного навантаження безпеки.
- Заголовок Параметри призначення.
- Заголовок верхнього рівня.

Вихідний вузол повинен слідувати цьому порядку, але вузли призначення повинні бути готові приймати їх у будь-якому порядку. Але існує їх рекомендований порядок [4, 5].

Порівняння заголовків IPv4 і IPv6 показує більш довгий заголовок, але менше число полів. А отже обробка заголовків значно спрощується. Усі додаткові параметри містяться і обробляються заголовками розширення.

1.9 Моніторинг комп'ютерних мереж IPv6

Традиційні підходи моніторингу, як правило, не застосовуються до трафіку IPv6 через тимчасові адреси, різні типи інкапсуляції IPv6 через IPv4, не єдине відображення між адресами каналів передачі даних та IP-адресами, тунелювання тощо. Таким чином, потрібні нові технології, що замінять існуючі інструменти.

Ще одним завданням моніторингу IPv6 є тунелювання IPv6 над IPv4. Тунелі інкапсулюють дані додатків у протоколи тунелювання, які мають різні IP-заголовки і порти, щоб пакети могли обійти правила брандмауера. Реальний перехід до IPv6 може тривати місяцями або роками, тому моніторинг тунельного трафіку дійсно необхідний для виявлення станцій, які можуть бути потенційно джерелами неконтрольованого трафіку користувачів.

Архітектура моніторингу IPv4 та IPv6, як і діяльність користувачів в мережі IPv6, може бути ідентифікована і зареєстрована навіть при використанні тимчасових адрес IPv6 або тунельного підключення. Запропонована та розроблена в даній роботі система моніторингу може бути розгорнута як в середовищі IPv4, так і в середовищі IPv6.

Порівняно з протоколом четвертої версії, для шостої версії існує дуже мала кількість робіт або досліджень, які обговорюють практичні питання моніторингу протоколу IPv6, такі як унікальність IPv6-адресності, проблеми тунелювання тощо. Автори [8] обговорюють проблеми безпеки IPv6 у порівнянні з загрозами IPv4. Вони перераховують автоконфігурацію, DoS-атаки на протокол ND (Neighbor Discovery) і труднощі з фільтрацією пакетів. Їх підхід зосереджується скоріше на огляді безпеки, а не на моніторингу та відстеженні діяльності користувачів. Звіт NIST [8] (National Institute of

Standards and Technology), здається, є найбільш обґрунтованим дослідженням аспектів безпеки IPv6 у практичному розгортанні з підказками для мережевих адміністраторів. Звіт охоплює широку область IPv6, включаючи питання безпеки, тунелювання, перекладу та нові протоколи. Звіт підсумовує різні методи та підказки для безпечного розгортання IPv6. Проте, кілька представлених методів, таких як Secure ND (SEND), ще не реалізовані.

2. ОБГРУНТУВАННЯ ВИБОРУ МЕТОДІВ РЕАЛІЗАЦІЇ

Розглянемо основні можливості розробленого програмного сервісу та функції, які він реалізує.

2.1 Тунелювання IPv6 над IPv4

Тунелювання - це технологія переходу [9], яка з'єднує сайти IPv6 через інфраструктуру IPv4. Маршрутизатори, брандмауери та пристрої безпеки на межі корпоративної мережі можуть бути не здатні технічно перевіряти інкапсульоване корисне навантаження IPv6 в межах пакетів IPv4, що надходять в або виходять з мережі. Часто застосовуються три тунельні підходи – 6to4 Teredo і ISATAP [9]. Всі ці три механізми тунелювання за замовчуванням включені на Windows 7 і вище. Таким чином, IPv4 може тунелювати трафік IPv6 без контролю безпеки, який може порушувати нормальну фільтрацію контролю доступу. Проблеми з тунелювання в IPv6 можна розглядати як подібні до тунелювання VPN в IPv4. Однак є деякі відмінності, VPN в цілому використовуються для підключення користувача з Інтернету до його мережі. Зазвичай потрібні логіні і пароль, тобто кінцева точка тунелю знаходиться під контролем адміністратора мережі. З іншого боку, тунелі IPv6 створюються зсередини мережі для підключення до загальнодоступного Інтернету. Це означає, що багато пристроїв, які, як правило, будуть приховані і захищені, можуть бути доступні в мережі через загальнодоступну адресу IPv6, призначену механізмом тунелю. Тунелі IPv6 також створюються автоматично без використання користувача. Наприклад, шлюз блокує весь вихідний трафік SMTP для попередження спаму. Дозволено пропускати лише трафік з авторизованих серверів SMTP. Оскільки брандмауер не перевіряє корисне навантаження тунельних пакетів, таких як 6to4 або Teredo, хост може поширювати спам, інкапсульований в пакети протоколу IPv4 (тобто, тунель 6to4) або UDP (тобто тунель Teredo).

2.2 Автоматична конфігурація тимчасових адрес IPv6.

Це нова функція IPv6, яка дозволяє самому вузлу самостійно генерувати адресу IPv6. Для забезпечення конфіденційності користувача, замість ідентифікатора EUI-64 віддають перевагу IPv6-адресам з випадково сформованими ідентифікаторами інтерфейсу 64 біт (так звані конфіденційні адреси). Стандарт RFC 4941 [9] визначає, як генерувати і змінювати тимчасові адреси. Важливою вимогою є те, що послідовність тимчасових генерованих адрес на інтерфейсі повинна бути абсолютно випадковою. Проте ця вимога суперечить необхідності ідентифікації зловмисного користувача. Приватні тимчасові адреси знищують унікальну ідентифікацію користувача/хоста, що підключається до служби, як це було для IPv4. Це впливає на ведення журналу і не дозволяє адміністраторам правильно відстежувати, які користувачі отримують доступ до цих служб. Поточна реалізація в системі Windows дозволяє за замовчуванням використовувати розширення приватності. Приклад: IPv6-адреса, призначена для хосту, наприклад. 2001: 718: 802: c0b1 :: 1. Хост має включене розширення приватності, тому генерує випадкову адресу, наприклад, 2001: 718: 802: c0b1: a197: 8afe: 5fe2: 5106, і це використовується для зв'язку замість призначеної 2001: 718: 802: c0b1 :: 1. Адреса є тимчасовою, тому через кілька годин генерується нова випадкова адреса.

2.3 Системи для моніторингу

2.3.1 Моніторинг у IPv4

На сьогодні провайдери ідентифікують свої хости на основі IPv4-адрес хоста. Зазвичай ISP (Internet Service Provider) має центральну систему керування мережею NMS(Network Management Station), яка збирає статистику мережі, включаючи список користувачів з зареєстрованими адресами IPv4 і MAC. MAC-адреса використовується в конфігурації DHCP для призначення відповідної адреси IPv4. Зареєстровані MAC-адреси разом з системними

журналами сервера DHCPv4 і даними з сервера Radius достатньо для однозначної ідентифікації користувача на основі адреси IPv4.

2.3.2Адміністрування адрес IPv6

Моніторинг користувачів трафіку IPv6 є більш складним. Адреса IPv6 більше не є унікальним ідентифікатором, як це було з адресою IPv4. Причиною цього є тимчасовість адрес. Існує два способи призначення адрес IPv6. З використанням DHCPv6 та конфігурація без статусу.

Конфігурація з протоколом IPv6 використовує DHCPv6 [11] для надання адрес IPv6 та інших параметрів конфігурації. На жаль, існує декілька причин обмеження DHCPv6, які не можуть бути використані для адресації за допомогою стану. Основна причина полягає в тому, що адреса шлюза за замовчуванням не може бути отримана через DHCPv6, тому конфігура-ція без статусу також повинна бути розгорнута. Це призводить до того, що системи Windows використовують тимчасову адресу для зв'язку замість адреси, яка отримана через DHCPv6, оскільки тимчасові адреси мають більш високий пріоритет. DHCPv6 також не ідентифікує хости з адресою MAC як DHCPv4, але з унікальним ідентифікатором DHCP. Це не можна легко використовувати як ідентифікатор користувача.

Конфігурація IPv6 без статусу використовує повідомлення RA(Router advertisement) - маршрутизація. Перша частина адреси IPv6 - префікс мережі - призначається за допомогою повідомлень RA разом із шлюзом за замовчуванням та іншими параметрами. Друга частина адреси IPv6 - ідентифікатор інтерфейсу - генерується з використанням EUI-64 (Extended Unique Identifier 64 bits) або розширень конфіденційності. Оскільки EUI-48 (Extended Unique Identifier 48 bits), що генерується з розширеннями конфіденційності, має більш високий пріоритет, ніж EUI-64, він також не може використовуватися як унікальний ідентифікатор.

Таким чином, жодна конфігурація не забезпечує унікальний ідентифікатор, необхідний для ідентифікації користувача. Це може бути досягнуто тільки шляхом комбінації декількох методів.

Проте, для конкретного хоста можна відслідковувати одразу декілька параметрів, а саме:

- його доступність;
- ping (час відгуку);
- набір та налаштування сервісів, розміщених на цьому хості;
- приблизне фізичне розташування хосту;
- стан вузлів на маршруті передачі пакетів.

Доступність на час відгуку хосту перевіряється за допомогою протоколу ICMPv6. Він повідомляє про помилки, якщо пакети не можуть бути оброблені належним чином і надсилає інформаційні повідомлення про стан мережі. Наприклад, якщо маршрутизатор не може перенаправити пакет, оскільки він занадто великий, щоб бути розісланим в іншій мережі, він відправляє повідомлення ICMP на початковий хост. Вихідний хост може використовувати це повідомлення ICMP, щоб визначити кращий розмір пакета, а потім повторно надіслати дані. Для перевірки якості з’єднання використовуються повідомлення ICMP Echo Request і Echo Reply (табл. 2.1).

Таблиця 2.1 – Типи ICMPv6 повідомлень, що використовуються для перевірки з'єднання

Message number	**Message type**	**Description**
128 129	Echo Request Echo Reply	RFC 2463. Both used for the ping command.

ICMPv6 є набагато більш потужним, ніж ICMPv4, і містить нові функціональні можливості. Наприклад, функція протоколу управління групою Інтернет (IGMP), яка керує приналежністю адреси до multicast групи.

Neighbor discovery - це протокол, введений у IPv6, що використовує повідомлення ICMPv6 для того, щоб визначити адреси зв'язку для сусідів, прикріплених до однієї підмережі, виявивлення конфліктів адрес та багато інших функцій (таблиця 2.1), що у IPv4 виконували протоколи ARP, ICMP, IRDP (Internet Router Discovery Protocol) и Router Redirect.

Таблиця 2.2 – Типи ICMPv6 повідомлень

Message number	**Message type**	**Description**
133 134 135 136 137	Router Solicitation Router Advertisement Neighbor Solicitation Neighbor Advertisement Redirect Message	RFC 2461. Used for neighbor discovery and autoconfiguration

У загальному вигляді ICMPv6 пакет складається з 4 і більше байтів (рис. 2.1). Першій байт містить у собі ідентифікатор типу повідомлення. Другий містить додаткову інформацію про помилку і його точний склад ґрунтується на першому полі, тобто на типі повідомлення (таблиця 2.2).

Наступні два байти займає контрольна сума. Далі ж міститься детальне повідомлення про помилку або просто тестові дані. Розмір цього поля повністю визначається типом повідомлення.

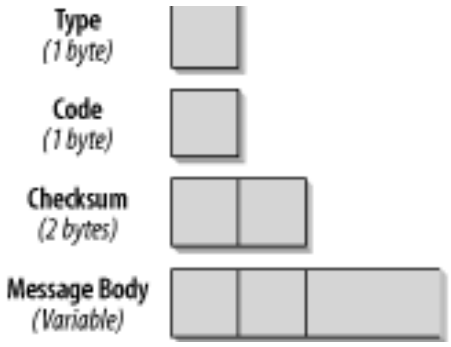

Рисунок 2.1 – Загальний вигляд ICMPv6

Також у програмі відловлюються та запам'ятовуються різні типи помилок (таблиця 2.3), що можуть виникати при визначення якості з'єднання з віддаленим сервером\хостом.

Таблиця 2.3 – Основні типи помилок у ICMPv6

Message number	**Message type**	**Code field**
1	Destination Unreachable	0 = немає маршруту до пункту призначення 1 = з’єднання адміністративно заборонено 2 = за межами адреси джерела 3 = адреса недоступна 4 = порт недоступний
2	Packet Too Big	Поле коду встановлюється 0 (нулем) відправником і ігнорується приймачем
3	Time Exceeded	0 = при передачі закінчився ліміт переходів з роутера до роутера 1 = перевищено час збору пакету

2.3.3 Перевірка типу сервісів

За допомогою пакетів цього протоколу в програмі також реалізовано перевірку типів сервісів, запущених на даному хості, тобто це дає можливиість дізнатися, яку саме функцію виконує пристрій за даною адресою. Це можна визначити за допомогою перевірки доступності та відгуку з стандартного переліку портів.

SMTP – 25

SMTP відомий як Простий протокол передачі пошти. Це пов'язано з TCP-портом № 25. Основна мета цього протоколу полягає в тому, щоб переконатися, що повідомлення електронної пошти передаються по мережі безпечно.

HTTP – 80

Порт 80 пов'язаний з протоколом HTTP(Hypertext Transfer Protocol) ы використовує TCP для доступу до мережі. Це один з найвідоміших і широко використовуваних портів у світі. Основна мета порту 80 - дозволити браузеру підключатися до веб-сторінок в Інтернеті.

HTTPS – 443

Порт - 443 також пов'язаний з протоколом TCP. Порт 444 HTTPS також дозволяє підключатися до Інтернету, встановлюючи з'єднання між веб-сторінками та браузером, що дозволяє підключатися до Всесвітньої павутини. Проте, цей порт має додаткові функції безпеки, яких не має у HTTP. Цей порт призначений для встановлення захищених з'єднань, щоб переконатися, що дані передаються через захищені канали та інші ресурси мережі.

FTP - 20, 21

Метою FTP (File Transfer Protocol) є передача файлів через Інтернет. Він, в основному, визначає всі правила, яких необхідно дотримуватися під час передачі даних.

Порт 20 виконує завдання пересилання та передачі даних. Він приймає на себе завдання передачі даних, коли знаходиться в активному режимі.

Порт 21 виконує завдання сигналізації для FTP. Він прослуховує всі команди і забезпечує управління потоком даних.

TELNET, 23

Порт 23 TELNET підпадає під категорію протоколів TCP. Її основною функцією є встановлення зв'язку між сервером і віддаленим комп'ютером. Він встановлює з'єднання після затвердження методу аутентифікації.

IMAP, 143

IMAP (Internet Message Access Protocol) - це протокол доступу до Інтернет-повідомлень. Основною метою цього порту є отримання електронних листів з віддаленого сервера без необхідності завантажувати електронну пошту. Користувач має право на доступ до електронної пошти з будь-якого місця. Після підключення до серверу та проходження аутентифікації відкривається можливість переглянути електронну пошту.

RDP 3389

RDP(Remote Desktop Protocol) також відомий як «Протокол для віддаленого робочого столу». Він працює через порт 3389 протоколу TCP. Цей порт був розроблений Microsoft. Він дає змогу встановити з'єднання з віддаленим комп'ютером. За допомогою цього з'єднання користувач отримуєте можливість управляти робочим столом цього віддаленого комп'ютера.

2.4 Python та Scrapy

Через таку велику кількість запитів до мережі Інтернет і складність їх формування, було вибрано мову програмування Python. Завдяки своїй гнучкій типізації вона дозволяє легко оперувати великими масивами даних, не розраховуючи кількість пам’яті, що необхідна для їх зберігання та обробки.

Також ця мова програмування відома великою кількістю бібліотек для обробки даних та роботи з мережею, що дозволяє максимально кастомізувати роботу з комп'ютерною мережею.

Для формування мережних пакетів була використана бібліотека Scrapy. В ній міститься велика кількість готових шаблонів готових пакетів. Наприклад, за набором таких команд буде сформовано пакет наступного змісту (рис. 2.2 - 2.3).

Бібліотека також підтримує так званий «сніффінг » пакетів, тобто вона дозволяє відловлювати, розбирати на складові та аналізувати будь-які пакети, що надходять на комп'ютер. Scrapy дозволяє створювати на основі свого функціоналу повноцінний кастомізований сніффер зі своїм набором фільтрів та налаштувань.

```
a = ARP()
a.pdst="192.168.1.208"
a.hwsrc="11:11:11:11:11:11"
a.psrc="1.1.1.1"
a.hwdst="ff:ff:ff:ff:ff:ff"
a.display()
```

Рисунок 2.2 – Приклад програмного коду для формування пакету

```
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
  hwsrc= 11:11:11:11:11:11
  psrc= 1.1.1.1
  hwdst= ff:ff:ff:ff:ff:ff
  pdst= 192.168.1.208
```

Рисунок 2.3 – Приклад сформованого пакету

2.5 Django

Для створення веб-орієнтованого користувацького інтерфейсу використовується Python фреймворк під назвою Django. Django - це фреймворк, який має на меті швидке розгортання і розробку різноманітних веб-додатків, написаних на Python. Цей фреймворк має відкритий вихідний код. Власне, сам по собі цей фреймворк - бібліотека кодів, яка допомагає розробнику в побудові надійних, легко масштабованих веб-додатків. Django - найпопулярніший з широкої різноманітності фреймворків, доступних для розробників, які пишуть на Python.

Однак, є одне обмеження: деякі речі можуть бути зроблені одним і тільки одним способом. Існує можливість замінити певні модулі, але все одно деякий внутрішній функціонал повинен залишатися незмінним. З одної сторони ця особливість зменшує загальну гнучкість проекту, з іншої же сторони – економить багато часу, що дозволяє реалізувати проекти будь-якого типу за значно менші відрізки часу.

2.5.1 Формат зберігання даних

Для роботи з базою даних у Django створено спеціальні структури даних – «моделі». Модель – це клас, що дозволяє зберігати у собі будь яку кількість інформації та легко серилізується для збереження в базі даних за допомогою вбудованих засобів. Приклад такого класу представлено на рис. 2.4.

```
from django.db import models

class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

Рисунок 2.4 – Клас моделі

За допомогою вбудованих функціональних можливостей створить у базі даних MySql таблиця такого вигляду (рис. 2.5).

```
CREATE TABLE myapp_person (
    "id" serial NOT NULL PRIMARY KEY,
    "first_name" varchar(30) NOT NULL,
    "last_name" varchar(30) NOT NULL
);
```

Рисунок 2.5 – Відображення моделі у MySql

При створенні моделі бази даних для веб-орієнтованого додатку було використано такі поля.

AutoField

Використовується для зберігання ID. Зазвичай, його не використовують, бо для стандартних моделей воно додається автоматично.

BinaryField

Поле для зберігання бітового масиву або навіть файлу. У них в розробленій системі зберігаються пакети у неформатованому вигляді.

BooleanField - зберігає значення true/false.

CharField (maxlength)

Це поле зберігає символьні дані з визначенням максимального розміру. В них зберігаються дані, що приходять з пакетами.

DateTimeField

Використовується для зберігання часу надходження пакетів.

DecimalField

Десяткове значення з фіксованою точністю; в полях цього типу зберігається більшість десяткових значень параметрів перехопленого пакету.

2.5.2 Обробка даних

Наступний програмний рівень - “views”, що займається обробкою даних введених користувачем і безпосередньо заповненням полів моделей. Кожен окремий файл цього модуля представляє собою набір функцій для обробки різних типів веб-запитів (POST, GET, PUT, DELETE etc.). Для обробки кожного окремого типу запиту створюються клас виду (рис. 2.6).

```
class PostUpdate(LoginRequiredMixin, UserPassesTestMixin, UpdateView):
    model = Post
    fields = ['title', 'body']
    template_name = 'blog/create_post.html'
    login_url = reverse_lazy('login')

    def test_func(self):
        return Post.objects.get(id=self.kwargs['pk']).user == self.request.user
```

Рисунок 2.6 – Приклад класу для обробки запиту на зміну полів збережених даних

Саме у цих класах викликаються модулі роботи з мережею та обробки даних. Також у них, за рахунок наслідування класів або встановлення декораторів виконуються функції аутентифікації та інші так звані «middleware». Вони являють собою функції або класи, та навіть цілі модулі, що обробляють дані поточної сессії та можуть управляти доступом до ресурсів.

Введені користувачем дані спочатку будуть перевірені на валідність, а потім записані до моделі збереження клієнтського запиту. Після цього управління буде передано модулю по роботі з мережею, а з нього у модуль аналізу даних. Після форматування результатів роботи, вони будуть записані до своїх моделей даних та записані до бази даних.

2.5.3 Відображення даних

Дані в уже відформатованому вигляді будуть передані до наступного програмного модуля під назвою «templates». Цей модуль представляє собою набір динамічних HTML документів. Тобто, це веб-сторінки, що змінюються в залежності від переданих до них даних. Найрозповсюдженіший підхід заснований на шаблонах, які містять у собі бажані статичні частини виводу HTML коду, а також деякий спеціальний синтаксис, що дозволяє інтегрувати на сторінці необхідні дані.

Наприклад, на веб-сторінку, що відповідає за відображення даних роботи одного запиту, будуть передані усі зібрані дані з мережевого модуля та деякі статистичні дані, отримані після обробки.

3. РОЗРОБКА КОМПОНЕНТІВ СИСТЕМИ

3.1 Загальна структура програми

Розроблений веб-орієнтований додаток реалізує функції моніторингу та аналізу за допомогою Django Web Framework та Scrapy. Для зручності він містить чотири основних модуля, два підмодуля та саме веб-сервер (рис. 3.1).

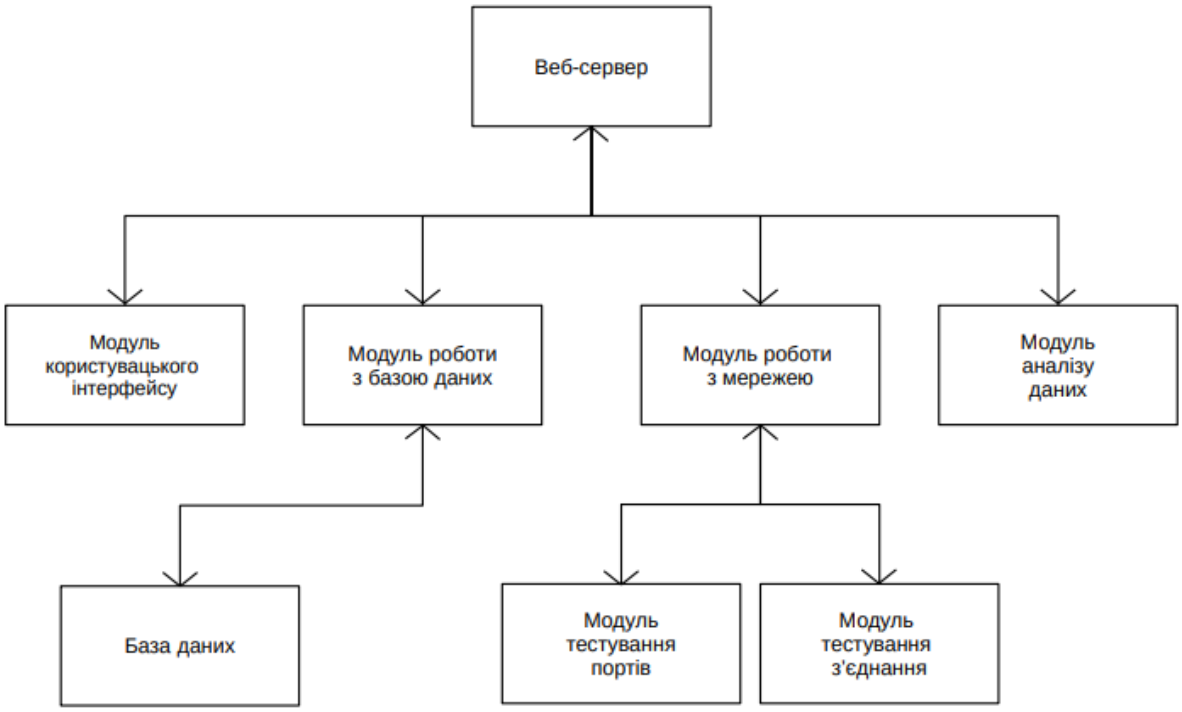

Рисунок 3.1 – Структурна схема

3.2 Модуль інтерфейсу користівача

Модуль користувацького інтерфейсу являє собою набір динамічних HTML шаблонів, доповнених спеціальним синтаксисом для взаємодій з веб-сервером. Він включає в себе:

- форма авторизації (логін);
- форма реєстрації;
- форма запиту на сканування додаткових сервісів;
- форму створення нового запиту;

- сторінку перегляду історії запитів;
- сторінку детального перегляду результатів конкретного запиту;
- форма редагування тексту результатів.

1) Форми авторизації

Розроблений додаток функціонує на основі веб-серверу, що може бути, при необхідності, розміщений на хостингу. Тобто, для отримання повноцінного доступу до функціональних можливостей і для забезпечення якомога більш широкого доступу до неї користувачів, програмний комплекс повинен бути розгорнутий на віддаленому сервері з присвоєним йому доменним ім'ям.

Для функціонування у режимі обслуговування значної кількості користувачів була також розроблена система авторизації та реєстрації. При створенні запиту, він зв'язується з користувачем. Надаючи йому (користувачу) індивідуальні права на редагування тексту результатів запиту та видалення його з історії.

Для створення нових користувачів, була розроблена форма реєстрації (рис 3.2) користувача. Після проходження процесу реєстрації користувач може переглядати історію відкритих користувацьких запитів, а після підтвердження імейл адреси – створювати свої.

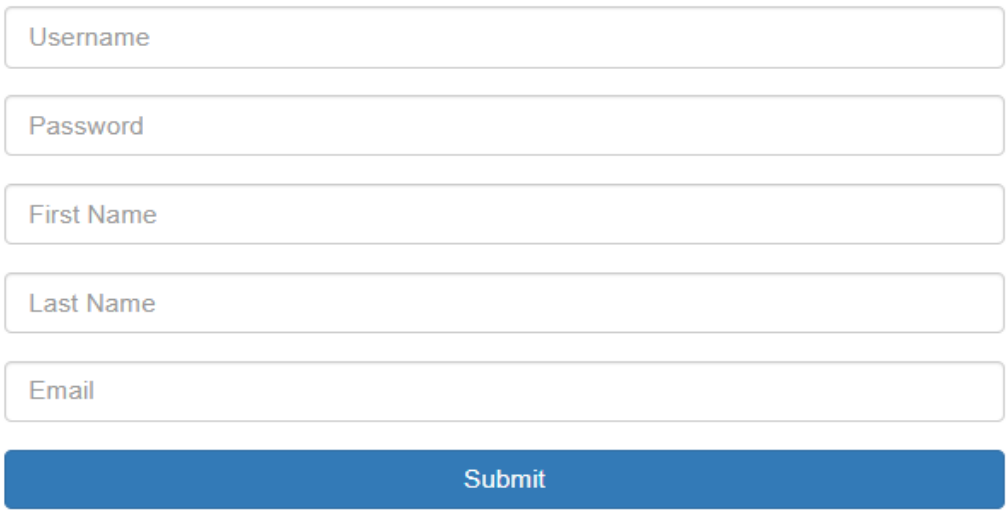

Рисунок 3.2 – Форма реєстрації користувача

Для авторизації уже зареєстрованих користувачів використовується проста форма, що потребує вводу двох полів: логіну і паролю (рис. 3.3).

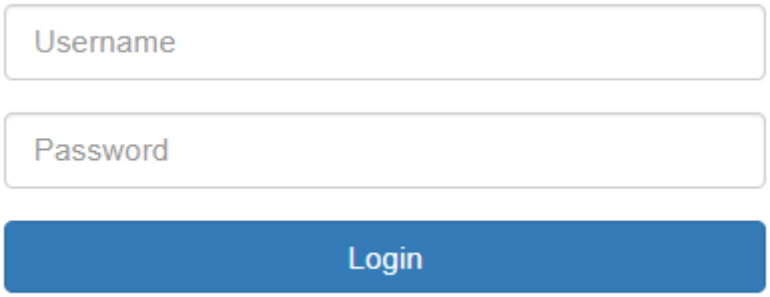

Рисунок 3.3 – Форма входу користувача

Обидві форми підтримують функції валідації (рис. 3.4-3.5) полів та захисту від спаму, шляхом генерації унікального токену. Наявність якого не дає можливості відправляти POST запити в обхід форми на сайті.

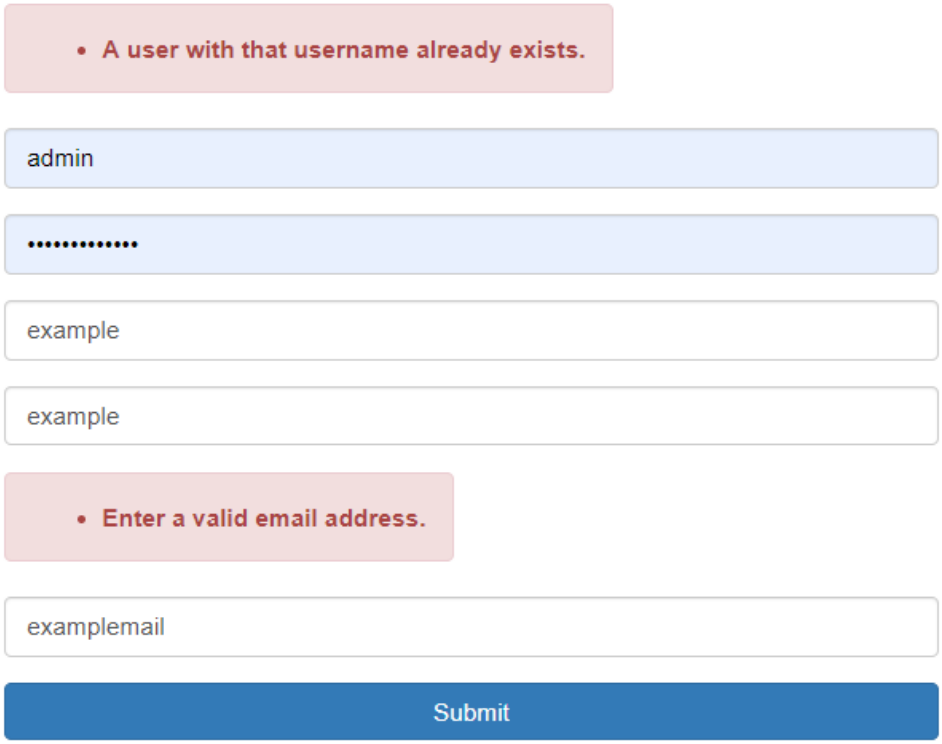

Рисунок 3.4 –Валідація форми реєстрації користувача

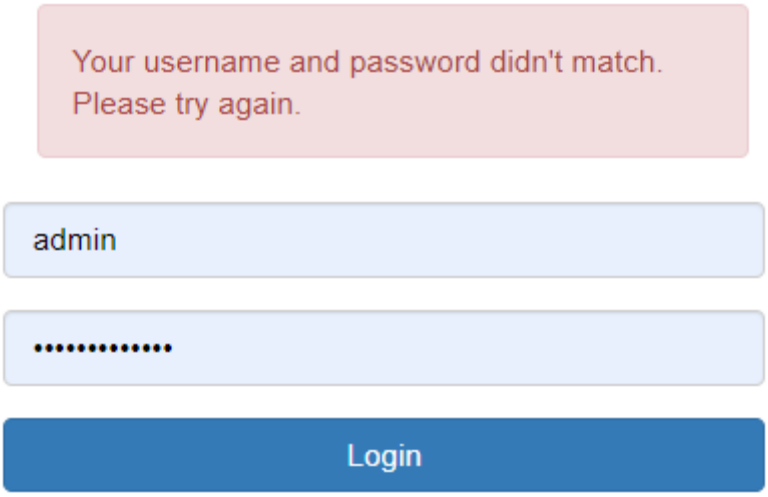

Рисунок 3.5 –Валідація форми входу користувача

3.3 Модуль роботи з базою даних

Цей модуль являє собою набір структур даних підготовлених до серіалізації та збереження до бази даних. Він включає в себе такі структури даних.

1) Модель зареєстрованого користувача

Ця модель зберігає у собі ім’я та прізвище зареєстрованого користувача, його імейл, аватар, логін та пароль у захешованому вигляді, а також список створених запитів, а також набір прав, що регулює можливість створення, видалення та перегляду користувачем історії запитів інших користувачів (рис. 3.6). Також дана модель містить поля для зберігання файлу зображення, тобто кожен користувач може призначити собі аватар профілю.

Набір прав користувача визначається цілим додатнім числом, тобто «рівнем доступу». «0» - означає права адміністратора, що може переглядати та виглядати абсолютно усі створені запити. «1» - підтвердженого користувача, що може створювати запити та переглядати історію своїх запитів та запити визначені для загального перегляду. Та останній рівень прав – «2», що являють собою права користувача з непідтвердженою електронної поштою. Вони включають у себе лише перегляд загальних даних.

```
class MyUser(models.Model):
    id = models.ForeignKey
    password = models.Password(max_length=20)
    username = models.CharField(max_length=20)
    email = models.CharField(max_length=20)
    surname = models.CharField(max_length=20)
    name = models.CharField(max_length=20)
    avatar = models.FileField
    rights = models.IntegerField
    reg_date = models.DateTimeField('register date', auto_now_add=True)
    requests = models.ForeignKeysArray(Request, on_delete=Cascade)

    def get_absolute_url(self):
        return reverse('auth:user', kwargs={'pk': self.pk})

    def __str__(self):
        return 'Useer: {username}'.format(username=self.username, requests=self.requests)
```

Рисунок 3.6 – Модель зареєстрованого користувача

2) Модель користувацького запиту

Модель користувацького запиту призначена для збереження даних запиту користувача та результатів обробки цього запиту. Також окремо зберігаються (рис. 3.7).

```
class UserRequest(models.Model):
    id = models.ForeignKey
    user = models.ForeignKeysArray(Myuser)
    ip = models.CharField(max_length=20)
    ports = models.IntegerArray
    result_data_connection_status = models.CharField(max_length=50)
    result_data_connection_details= models.CharField(max_length=150)
    result_data_servises = models.CharField(max_length=20)
    result_data_servises_details = models.CharField(max_length=20)
    packets = models.ByteArray(max_length=1000)
    req_date = models.DateTimeField('creation date', auto_now_add=True)

    def get_absolute_url(self):
        return reverse('request', kwargs={'pk': self.pk})

    def __str__(self):
        return 'UserRequest {date} by {user.name}\nSend to: {data}'.format(usern=self.user, data=self.ip, date=req_date)
```

Рисунок 3.7 – Модель для зберігання даних запиту

Вона містить у собі такі поля:

- поле даних про користувача, що зробив запит;
- поле вхідних даних;
- поле з номерами портів для перевірки;
- поле результуючих загальних даних про стан з'єднання;
- поле результуючих повних даних про стан з'єднання;
- поле результуючих даних про стан активних сервісів;
- поле, що зберігає у собі масив даних, отриманих при перехоплюванні пакетів.

3) Модель для зберігання відформатованих результатів запиту

Ця модель містить у собі результуючі дані, відформатовані у HTML форматі для подальшого відображення на сторінці. Також містить у собі посилання для зв'язку за моделями користувача та моделі запиту (рис. 3.8).

```
class Post(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    title = models.CharField(max_length=80)
    body = models.TextField(blank=True)
    pub_date = models.DateTimeField('date published', auto_now_add=True)

    def get_absolute_url(self):
        return reverse('blog:post', kwargs={'pk': self.pk})

    def __str__(self):
        return '"{title}" by {username}'.format(title=self.title,
                                                username=self.user.username)
```

Рисунок 3.8 – Модель для зберігання відформатованого HTML

3.4 Модуль роботи з мережею

Цей модуль містить у собі функції взаємодії розробленого веб-додатка з мережею, такі як сніфер мережевих пакетів, структури даних для зберігання стеку перехоплених пакетів та функції їх сортування та первинної обробки (рис 3.9).

Клас сніфера відловлює усі пакети, що надходять до системи за допомогою бібліотеки Scapy, та розподіляє їх по об'єктам сессій, що містять у собі черги пакетів, що були надіслані та чергу для зберігання пакетів, що надходять як відповідь.

```
class Sniffer(object):
    def __init__(self, our_ip):
        # self.file = open(FILE_LOGGER, "w")
        self.our_ip = our_ip
        self.current_packet = None
        self.sessions = {}

    def get_sessions(self):
        return self.sessions

    def set_session(self, packet, stemp, our_ip):
        self.sessions[stemp] = session.Session(packet, stemp, our_ip)

    def decide_stamp(self, three_tuple):
        if self.our_ip != str(three_tuple[0]):
            templ = three_tuple[1]
            three_tuple[0] = templ
            three_tuple[1] = three_tuple[0]
        return tuple(three_tuple)

    def update_next_packet(self):
        packet = s.sniff(count=1)  # filter = "tcp.len > 0",
        packet = packet[0]
        if s.IP not in packet:
            return
        ip_send = packet[s.IP].src
        ip_rec = packet[s.IP].dst
        if ip_send != self.our_ip and ip_rec != self.our_ip:
            return
        if make_stamp(packet) is not None:
            ip_send, ip_rec, protocol = make_stamp(packet)
            three_tuple = [ip_send, ip_rec, protocol]
            stamp = self.decide_stamp(three_tuple)
            if self.sessions.get(stamp) is None:
                threading.Thread(target=self.set_session, args=[packet, stamp, self.our_ip]).start()
            else:
                threading.Thread(target=self.sessions[stamp].update_session, args=[packet]).start()
                if self.sessions[stamp].got_fin is True:
                    to_add = self.sessions.pop(stamp)
                    sorted(to_add.income, key=key_func)
                    sorted(to_add.outcome, key=key_func)
                    sorted(to_add.combined, key=key_func)
                    lst.add(to_add)
        else:
            print("[Error]")
```

Рисунок 3.9 – Клас сніферу

Кожна окрема сесія оброблюється в ізольованому потоці. Вона створюється коли відправляться запит до мережі та чекає на відповідь. Сесія містить у собі час відправлення пакету з комп'ютера та час, коли надійшла відповідь, що дозволяє визначити точний час обробки і передачі запиту до віддаленого серверу (рис 3.10).

```
class Session(object):
    def __init__(self, pkt, session_info, our_ip):
        self.our_ip = our_ip
        self.lock = threading.Lock()

        if str(session_info[0]) == our_ip:
            self.income = [(pkt, 0)]
            self.outcome = []
        else:
            self.outcome = [(pkt, 0)]
            self.income = []
        self.combined = [(pkt, 0)]
        self.session_info = session_info
        self.start_time = pkt.time
        self.got_fin = False

    def update_session(self, pkt):
        time_now = pkt.time
        self.lock.acquire()
        self.combined += [(pkt, time_now - self.start_time)]

        if pkt[s.IP].src == self.our_ip:
            self.outcome += [(pkt, time_now - self.start_time)]
        else:
            self.income += [(pkt, time_now - self.start_time)]
        if s.TCP in pkt:
            self.got_fin = check_if_got_fin(pkt)
        else:
            self.got_fin = True
        self.lock.release()
```

Рисунок 3.10 – Клас сессії

3.5 Модуль тестування портів

Цей модуль призначений для тестування віддаленого серверу (хоста) на наявність на ньому певного набору запущених сервісів. Це дозволяє визначити призначення віддаленого пристрою та набір його можливостей. Даний модуль

також побудований на принципі багатопоточності, що дозволяє пришвидшити загальний час обробки запиту у декілька разів (рис 3.11).

```
print_lock = threading.Lock()
def portscan(port):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        con = s.connect((target,port))
        with print_lock:
            print('port',port)
        con.close()
    except:
        pass

# The threader thread pulls an worker from the queue and processes it
def threader():
    while True:
        # gets an worker from the queue
        worker = q.get()
        # Run the example job with the avail worker in queue (thread)
        portscan(worker)
        # completed with the job
        q.task_done()

# Create the queue and threader
q = Queue()
# how many threads are we going to allow for
for x in range(30):
     t = threading.Thread(target=threader)
     # classifying as a daemon, so they will die when the main dies
     t.daemon = True
     # begins, must come after daemon definition
     t.start()
start = time.time()
# 100 jobs assigned.
for worker in range(1,100):
    q.put(worker)
# wait until the thread terminates.
q.join()
```

Рисунок 3.11 – Основний алгоритм модулю тестування сервісів

3.6 Модуль тестування з’єднання

Даний модуль відповідає за генерацію ICMP пакетів та їх розсилку. Пакет даних формується наступним чином (рис 3.12).

```
def make_packet(self):
    checksum = 0
    header = struct.pack(
        "!BBHHH", self.icmp_echo, 0, checksum, self.own_id, self.seq_number
    )
    pad_bytes = []
    start_val = 0x42
    for i in range(start_val, start_val + (self.packet_size-8)):
        pad_bytes += [(i & 0xff)]  # Keep chars in the 0-255 range
    data = bytearray(pad_bytes)
    checksum = self._checksum(header + data)
    header = struct.pack(
        "!BBHHH", self.icmp_echo, 0, checksum, self.own_id, self.seq_number
    )
    return header + data
```

Рисунок 3.11 – Формування ICMP пакетів

Після процессу створення пакетів вони через сокети відправляються на віддалений сервер (рис. 3.12).

```
for i in range(0, times):
    try:
        my_socket = self.make_socket()
    except socket.error as e:
        etype, evalue, etb = sys.exc_info()
        if e.errno == 1:
            msg = "{} - ICMP messages can only be send from processes running as root.".format(evalue)
        else:
            msg = str(evalue)
        self._echo_message(msg)
        response.messages.append(msg)
        response.ret_code = FAILED
        return response
    try:
        send_time = self.send(my_socket, dest_ip)
    except socket.error as e:
        msg = "General failure ({})".format(e.args[1])
        self._echo_message(msg)
        response.messages.append(msg)
        my_socket.close()
        return response
    if not send_time:
        response.ret_code = Ping.FAILED
        return response
    receive_time, packet_size, ip, ip_header, icmp_header = self.receive(my_socket)
    my_socket.close()
    delay = self._calc_delay(send_time, receive_time)
```

Рисунок 3.12 – Відправка ICMP пакетів

3.7 Модуль аналізу даних

Цей модуль призначено для обробки інформації з сесій та сбору статистичних даних (рис. 3.13).

```
class SessionFeatureExtractor(object):
    def __init__(self, session):
        self.session = session
        self.all_packets = session.combined
        self.in_pkts = session.income
        self.out_pkts = session.outcome
    def get_feat(self):
        if self.session.session_info[2] == "TCP":
            proto = 1
        else:
            proto = 0
        curr_features = 1
        n_features = 7
        num_small_packets_pkt_s = get_n_small(self.in_pkts)
        num_small_pkt_c = get_n_big(self.out_pkts)
        curr_features += 1
        cc_len_sec = get_lens_per_sec(self.in_pkts)
        curr_features += 1
        cl_len_sec = get_lens_per_sec(self.out_pkts)
        curr_features += 1
        avg_c2c, avg_s2c2s = get_delay_average(self.session, self.session.our_ip)  # use in_pkt
        curr_features += 2
        max_c, max_s = get_max_delay(self.session, self.session.our_ip)
        curr_features += 2
        return num_small_pkt_c, num_small_packets_pkt_s, cc_len_sec, cl_len_sec, avg_c2c, avg_s2c2s, max_c, max_s
    def get_delay_average(session, our_ip):
        cnt_A = 0
        cnt_B = 0
        delay_sum_A = 0
        delay_sum_B = 0
        curr = session.combined[0]
        for i in xrange(1, len(session.combined)):
            pkt_tuple = session.combined[i]
            prev = curr
            curr = pkt_tuple
            if prev[0][s.IP].src != our_ip and curr[0][s.IP].src == our_ip and i < len(session.combined) - 1:
                i += 1  # to skip the
                pkt_tuple = session.combined[i]
                prev = curr
                curr = pkt_tuple
                while prev[0][s.IP].src == our_ip and curr[0][s.IP].src == our_ip and i < len(session.combined) - 1:
                    delay_sum_A += (curr[1] - prev[1])
                    i += 1
                    pkt_tuple = session.combined[i]
                    prev = curr
                    curr = pkt_tuple
                    cnt_A += 1

                delay_sum_B += (curr[1] - prev[1])
                cnt_B += 1
        r1 = r2 = 0
        if cnt_A != 0:
            r1 = delay_sum_A / cnt_A
        if cnt_B != 0:
            r2 = delay_sum_B / cnt_B
        return r1, r2
```

Рисунок 3.13 – Обробка даних сесій

У модулі також присутні функції аналізу значень затримки на основі аналізу параметру TTL. Обробка інформації дає змогу припустити чому був маршрут. Наприклад, коли при зменшенні TTL зростає затримка, то все просто

– один із маршрутизаторів має переповнену чергу і пакет відправляється альтернативним маршрутом. Якщо TTL не змінюється, але при цьому зростає затримка, можна стверджувати про завантаженість шляхів передачі. Найцікавіша ж ситуація виникає, коли при зменшенні TTL зменшується затримка, що може означати, що було знайдено новий оптимальний маршрут або при великому завантаженні на основному шлюзі, пакет було перенаправлено на запасний шлюз і за рахунок зменшення часу очікування у черзі було значно зменшено затримку. Такі випадки можу виника лише у дійсно глобальних мережах з потужними маршрутизаторами, наприклад у магістральних провайдерів.

4. ТЕСТУВАННЯ КОМПОНЕНТІВ СИСТЕМИ

Для тестування системи необхідно впевнитися, що провайдер та роутер, що обслуговують сервер, на якому розгорнуто розроблений веб-орієнтований додаток, підтримують IPv6 з'єднання. Тобто необхідно переконатися, що роутер містить видану провайдером IPv6 адресу і сам сервер містить локальні, видані роутером, адреси шостої версії протоколу. Якщо ж з цим виникають проблеми, то одним із можливих рішень є використання спеціальних VPN. При правильному налаштуванні підключення до таких мереж весь IPv6 трафік буде тунельовано через мережу Інтернет до приватної мережі і уже з її серверів відправиться у вигляді справжніх IPv6 пакетів до адресата.

Після проведення усіх налаштувань потрібно лише зайти на доменне ім'я або ip-адресу, на якій розгорнуто веб-додаток (рис. 4.1).

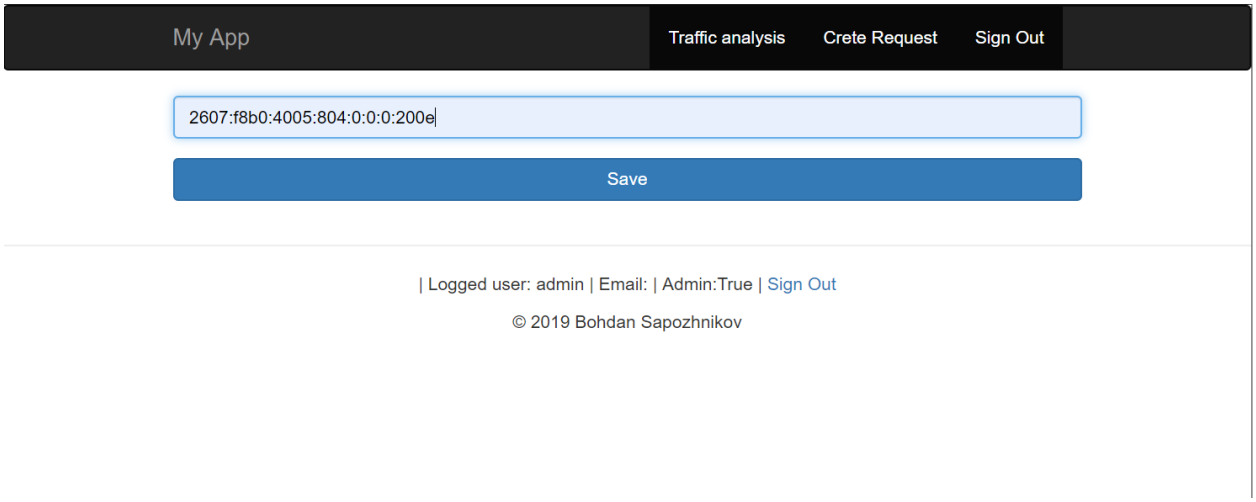

Рисунок 4.1 – Вікно вводу адреси

Після вводу даних, вони будуть валідовані. У випадку якщо була введена IP адреса, то вона одразу буде направлена в обробку, а якщо доменне ім'я - то буде віпдправлено ARP запит на отримання адреси.

Відбудеться затримка у 15-20 секунд (залежно від віддаленості хоста), після якої буде отримано дані та проведено їх аналіз. Після завершення обробки результатів вони будуть виведені на веб-сторінку у такому форматі (рис 4.2).

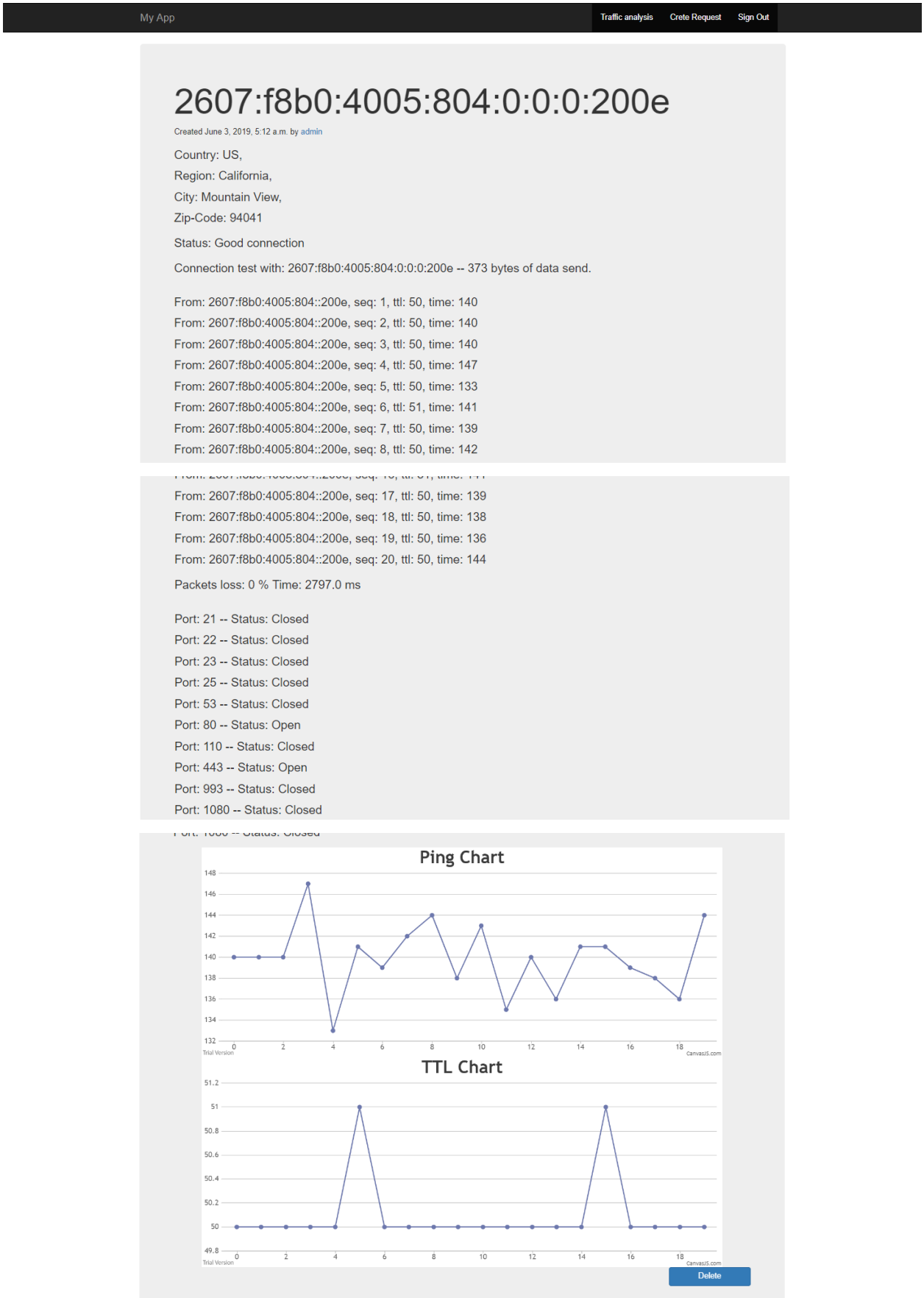

Рисунок 4.4 Приклад виводу результатів роботи

ВИСНОВКИ

Розроблений веб-орієнтований додаток дозволяє збирати, обробляти та зберігати дані про стан мережі, яка функціонує на основі протоколу IPv6. Також він надає інформацію про те, які служби або сервіси розміщені на конкретних віддалених хостах. Це надає можливість ,проаналізувавши отримані дані, ідентифікувати тип та призначення даного хоста.

Отже, розроблений додаток надає можливість досліджувати якість з'єднання у мережах нового(шостого) покоління, що, безперечно, є дуже важливим, адже мережі, що використовують протокол IPv6, тільки починають свій розвиток, як у нашій країні, так і світі загалом. Тож важливим є знати наскільки стабільним і якісним є зв'язок з потрібними віддаленими ресурсами.

Веб-орієнтований додаток створено для підтримки лабораторних робіт курсу «Комп'ютерні мережі». Створення цього програмного комплексу дозволяє засвоїти на практиці та поглибити знання з даного курсу.

Переваги: можливість застосовувати на будь-якій операційних системах та пристроях, що підтримують браузер; простий та зрозумілий веб-інтерфейс; наочне подання отриманих результатів.

Недоліки: велике навантаження на мережеве з'єднання серверу; необхідність використання веб-хостингу для повноцінного використання.

Можливі модифікації:

- створення API та мобільного додатку для роботи з ним;
- додавання нових модулів обробки даних.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Орлова М.М. Конспект лекцій з курсу «Комп'ютерні мережі».
2. Computer Network | Layers of OSI Model [Електронний ресурс] https://www.geeksforgeeks.org/layers-osi-model/
3. IPv4 Packet Structure [Електронний ресурс] http://thirdinternet.com/ipv4-packet-structure/
4. Internet Protocol version 6 (IPv6) [Електронний ресурс] https://www.tenouk.com/internetprotocolversion6ipv6.htm.
5. IETF. The premier Internet standards organization documents [Електронний ресурс] https://www.ietf.org/standards
6. Traffic Classification and Quality of Service (QoS) [Електронний ресурс] https://tools.ietf.org/html/rfc5777
7. A Brief History of IPv4 [Електронний ресурс] http://ipv4marketgroup.com/a-brief-history-of-ipv4/
8. NIST IPv6 Profile [Електронний ресурс] https://www.nist.gov/sites/default/files/documents/2018/07/12/draft-nist-sp-500-267ar1.pdf
9. "Все технологии туннелирования IPv6 понятным языком" [Електронний ресурс] https://habr.com/ru/post/207562/